Escuela Politécnica Superior de Elche

SISTEMAS INFORMÁTICOS EN TIEMPO REAL

2º Ingeniería Industrial

PRÁCTICAS DE PROGRAMACIÓN DE MINIROBOTS

Taller 1. Introducción a Interactive C (IC)

Luis Miguel Jiménez

Rafael Puerto

Departamento de Ingeniería
Área de Ingeniería de Sistemas y Automática

1 OBJETIVO

El objetivo de este documento es mostrar una introducción a la programación de robots utilizando el entorno IC sobre un microcontrolador Motorola 68HC11. Se recomienda consultar el servidor web (http://lorca.umh.es/isa/es/temas/minirobots) donde se encuentra documentación adicional, así como el libro "*Mobile Robots: Inspiration to Implementation*" referencia básica. Así mismo se recomienda visitar la página web del MIT donde se pueden encontrar grupos de usuarios utilizando la misma plataforma.

Se presentarán los primeros pasos en la realización de una aplicación utilizando el robot RugWarrior Pro, así como el uso básico del entorno de programación, descarga de programas, ejecución y depuración.

2 MATERIAL EMPLEADO

La práctica se realizará en grupos de tres personas, disponiendo de un PC con S.O. Windows 95/98, el entorno de desarrollo ICWin, y un Robot RugWarrior Pro. La documentación se puede encontrar en el servidor http://lorca.umh.es/isa/es/temas/minirobots

2.1 ¿Qué es IC?

Interactive C (**IC** a partir de ahora) es un entorno de desarrollo para microntroladores de la familia Motorola **68HC11**. Se trata de un microcontrolador basado en una CPU de 8 bits que incorpora un conjunto de módulos para entrada salida temporizada muy eficientes para el diseño de aplicaciones de tiempo real. Permite así mismo integrar un sistema completo con muy pocos chips adicionales, permitiendo diseñar tarjetas de control de reducidas dimensiones y bajo consumo.

Se trata evidentemente de un micro muy antiguo y de escasa potencia comparado con las modernas CPUs de los PCs actuales, pero atesora virtudes importantes que le ha permitido seguir en uso en multitud de sistemas empotrados: es un dispositivo muy sencillo de programar, es muy robusto y determinista, consume poca energía, pero sobre todo, integra en su hardware las funciones básicas de gestión de entradas y salidas para manejar la mayoría de sensores y actuadores (cualidad fundamental en un sistema empotrado)

Como hemos comentado, la primera característica de IC es que está orientado a realizar programas sobre el uC 68HC11. La segunda y de gran importancia es que se trata de un entorno que se diseñó pensando en el control de pequeños robots, por lo que incorpora de forma nativa muchas funcionalidades necesarias para accionar motores eléctricos, servomotores, leer sensores de proximidad, distancia, etc.

Otro aspecto importante es que incorpora un planificador multitarea, permitiendo ejecutar de forma concurrente varios procesos.

IC fue desarrollado por investigadores del Media Lab del MIT, el centro más importante del mundo en Inteligencia Artificial. El objetivo era disponer de una herramienta sencilla que permitiera a los estudiantes integrar conceptos de inteligencia artificial en pequeños robots. Son especialmente famosos los concursos de robots que tuvieron su origen en este centro y que utilizaban IC como herramienta básica. Estos concursos tenían como objetivo asimilar, por parte del estudiante, metodologías de diseño en ingeniería mediante el autoaprendizaje.

El lenguaje base utilizado por IC es un derivado del **lenguaje C**. El lenguaje C es especialmente potente por su capacidad de gestionar recursos de bajo nivel (necesarios para programar microcontroladores), a la vez de poder realizar una programación estructurada de alto nivel. La implementación de C utilizada en IC no es completa, eliminándose aquellos aspectos que se consideraron innecesarios (recordemos que los microcontroladores tienen pocos recursos de memoria por lo que deben ser bien aprovechados). En todo caso incorpora toda la funcionalidad básica con todas las estructuras de control de alto nivel.

A parte del lenguaje C, el entorno IC permite incorporar de forma sencilla código **ensamblador** para aquellas tareas más críticas (gestión de interrupciones..), permitiendo intercomunicar el código C y el código ensamblador mediante variables globales.

El entorno IC está formado por los siguientes módulos:

- ***Entorno de desarrollo y depuración (IDE):*** que se ejecuta sobre un PC (DOS, Windows) o una estación UNIX: incluye un editor, un compilador de C, ensamblador y un interprete para depuración. Incorpora asimismo funciones para descargar el software sobre el microcontrolador por medio de un puerto RS-232, así como un depurador que nos permite evaluar las variables y el estado del microcontrolador.

- ***Sistema operativo para el microcontrolador 68H11 (pcode):*** gestiona la comunicación con el PC, integra un depurador interactivo controlable desde el PC, permite la descarga de aplicaciones y, sobre todo, integra las funciones básicas de control de todas las entradas salidas y el planificador multitarea.

IC dispone de dos modos básicos de programación:

- **Interprete**: permite ejecutar comandos y líneas de código desde el entorno IDE. Nos permite probar funciones o comprobar el estado de las variables de forma interactiva.

- **Compilador**: lee un fichero con la aplicación, la compila en ***pseudo-código*** y lo transfiere al microcontrolador. El compilador no genera código máquina para una CPU concreta sino que utiliza una CPU virtual más sencilla. Este ***pseudo-código*** es interpretado por IC para su ejecución en el 68HC11 (esquema similar al utilizado por el lenguaje **Java**).

  Cada vez que se descarga un fichero, éste queda almacenado en la memoria del microcontrolador hasta que sea borrado o se agoten las baterías, por lo que

podemos ir incorporando librerías de funciones adiciones a las soportadas por el S.O. El S.O. tiene un registro de todas las funciones C cargadas.

El microcontrolador solo puede almacenar un programa (función *main()*) que es el se ejecuta cada vez que se enciende o inicializa el equipo. En cambio, podemos cargar tantos ficheros con librerías de funciones como quepan en la memoria RAM. La memoria disponible es de 32 KB, de los cuales aproximadamente la mitad (16 KB) son utilizadas por el código del *pcode* (S.O). El resto, entorno a 16 KB, están disponibles para los datos (variables) y el código (C o ensamblador) de nuestra aplicación.

2.2 ¿Qué es el RugWarrior Pro?

El RugWarrior Pro es un pequeño robot educacional basado en el microcontrolador Motorla 68HC11. Presentado en el libro de Joseph Jones y Anita Flyn "*Mobile Robots: Inspiration to Implementation*" como un ejercicio práctico de integración de las tecnologías asociadas al diseño de robots de forma asequible, y fácilmente reproducible.

Está constituido por tres partes principales (figura 1):

1. Un sistema de **accionamiento** formado por dos motores de corriente continua con reductora Los motores accionan sendas ruedas que conforman una configuración de tipo diferencial
2. Un conjunto de **sensores**: un micrófono, un encoder incremental para cada rueda, dos fotocélulas, un sensor de distancia infrarroja junto a dos emisores y tres interruptores para detectar colisiones.
3. Una tarjeta **microcontroladora** basada en el MC68HC11 con 32Kb de RAM, puerto RS-232, una pantalla LCD, altavoz, así como otros componentes para acondicionamiento de señal para los sensores disponibles. Este microcontrolador dispone de tres puertos (con 8 E/S cada uno) de entradas salidas (digitales y analógicas), uno de ellos con control temporizado. Buena parte de ellos están ocupados para la gestión de los sensores y actuadores disponibles, pero existen varios canales de entrada/salida disponibles así como un puerto de expansión (RugIO).

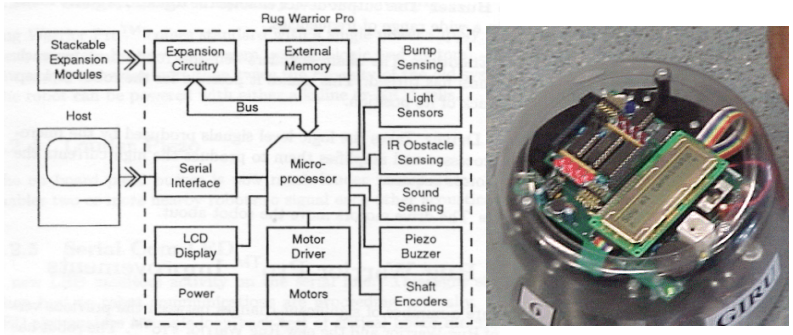


Figura 1 Diagrama de bloques del Rug Warrior Pro

Para profundizar en los detalles de su construcción se recomienda leer el libro comentado o la guía de montaje disponible en el servidor web.

2.3 El microcontrolador Motorola 68HC11

El cerebro del robot es la tarjeta controladora equipada con el microcontrolador MC68HC11. La figura 2 muestra el diagrama de bloques de este circuito integrado:
Incorpora los siguientes elementos:

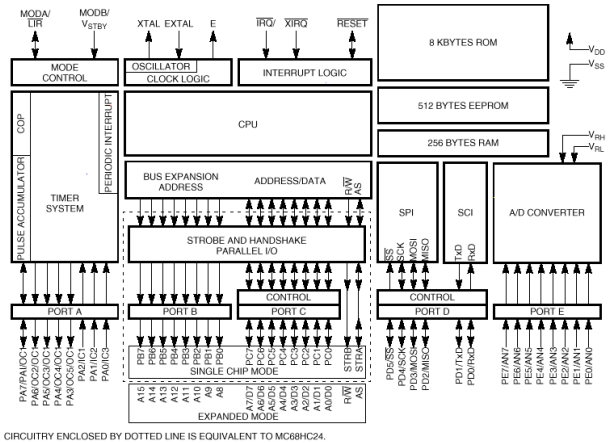


Figura 2 Diagrama de bloques del MC68HC11

- CPU de 8 bits compatible 6800 equipado con dos registros acumuladores de 8 bits y 4 registros de 16 bits (2 índices, Contador de Programa y Puntero del la Pila).
- Memoria RAM interna de 256 bytes (externamente se pueden disponer de memoria adicional)
- Memoria ROM/EPROM 8 Kbytes
- Bus de Datos de 8 bits y bus de direcciones 16 bits multiplexados en los puertos **B** y **C**.
- Puerto de comunicaciones serie asíncrona SCI (RS-232 con niveles 0-5 V)
- Puerto de comunicaciones serie síncrona de alta velocidad SPI
- Puerto **E** de entradas analógicas (8 entradas multiplexadas)
- Puerto **A** con 8 entradas/salidas temporizadas: 3 entradas con captura de tiempo (*Input Capture*), 4 salidas controladas por temporizador (*Output Compare*), 1 entrada/salida con acumulador de pulsos de 8 bits.

Cualquier señal no utilizada para su uso primario puede ser utilizada como una entrada o salida digital.

Para más información sobre este microcontrolador consultar el manual de referencia disponible en formato pdf en la página de documentación del servidor web.

3 UTILIZACIÓN DEL ENTORNO DE DESARROLLO IC

Se presenta en este apartado una introducción al manejo del entorno IC para Windows utilizando las funciones principales de compilación, transferencia por puerto serie, y los principios básicos de programación y depuración.

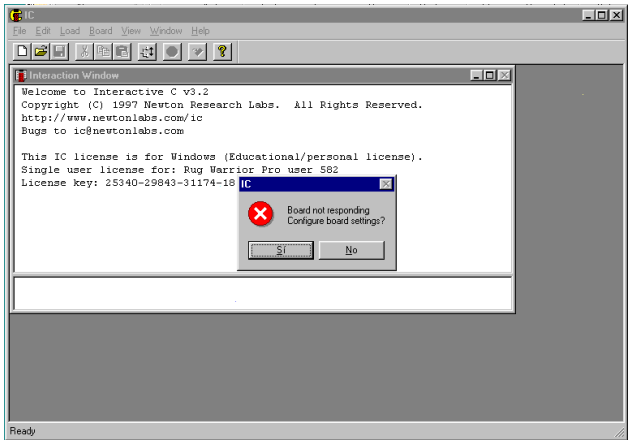
3.1 Arrancando IC

Utilizaremos la versión del entorno IC para Windows que está instalada en los equipos del laboratorio de Automática (*Nota: el software está disponible en el servidor web*).

1) Pincharemos sobre el icono **ICWin** o bien ejecutaremos la aplicación (**c:\ic\icwin.exe**).

   ***Nota**: en algunos equipos el programa puede estar ubicado en (c:\icRW\icwin.exe)*

   Al ejecutar ICWin aparecerá una ventana como la siguiente:



   Todavía no hemos conectado el robot al PC por lo que nos indica que no es capaz de detectar la tarjeta.

2) Conectaremos el robot al PC a través del cable serie suministrado (el conector del robot es de tipo telefónico).

3) Vamos a considerar que la memoria RAM está vacía (se ha quedando sin baterias) o que tenemos que reinstalar el S.O. Pondremos para ello la tarjeta en un modo especial de descarga (*Download*). Para ello buscaremos el interruptor ubicado junto al conector serie (figura 3), éste conmutador tiene tres posiciones: centro: ***apagado***, superior ***Run***, e **inferior**: ***download***. Lo pondremos en la posición inferior (etiqueta *download*).

En este modo el 68HC11 ejecuta un pequeño programa que lleva en ROM y que se encarga de leer 256 bytes del puerto serie copiándolos en la memoria RAM interna. Una vez copiados ejecuta el código descargado.

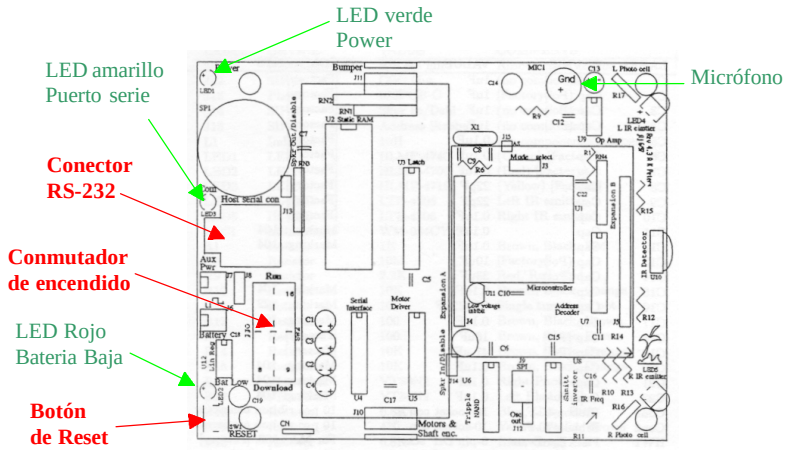
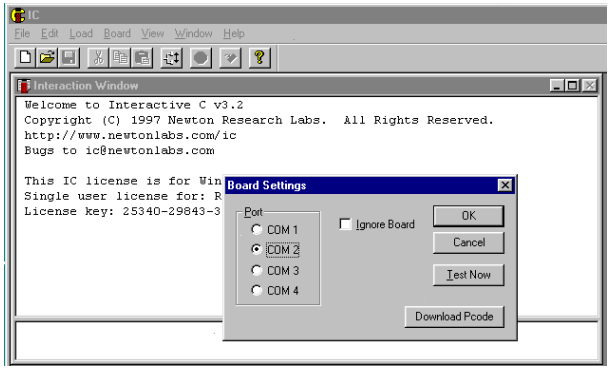
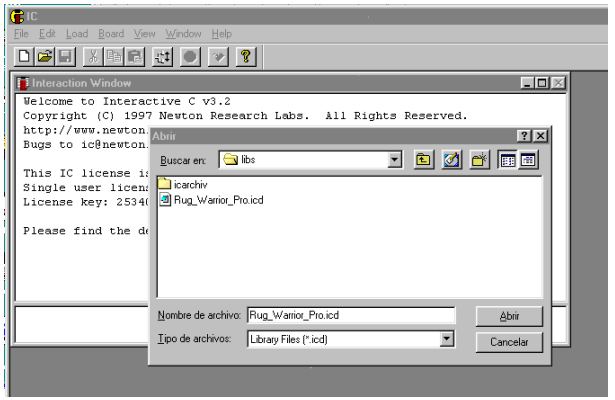


Figura 3 Distribución de componentes de la tarjeta controladora

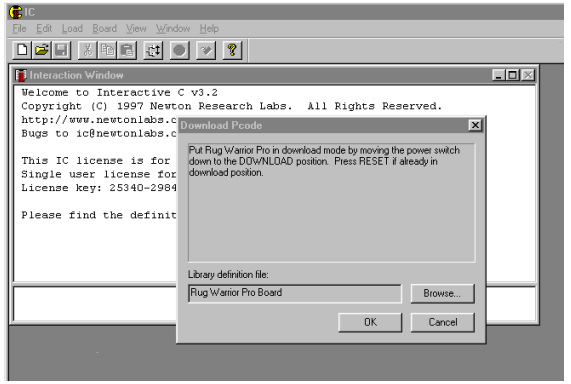
4) Volvemos al PC y seleccionamos el botón ***Si*** confirmando que queremos configurar la tarjeta. Nos aparecerá una ventana donde podemos configurar la conexión. El puerto utilizado suele ser **COM2**.



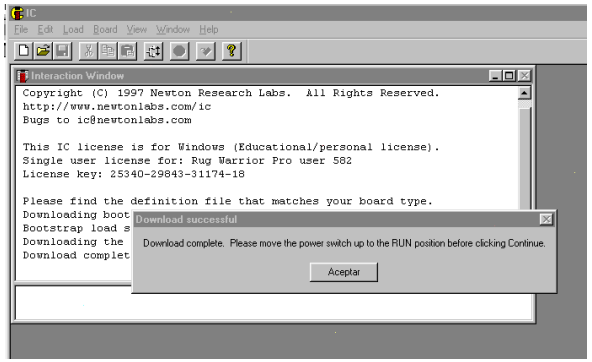
5) Pinchamos en el botón ***Download Pcode*** (*Pcode* es el S.O.). Nos sale otra ventana de diálogo donde debemos seleccionar el fichero de configuración (***icd***) del robot. Este fichero almacena las librerías específicas de manejo de sensores y accionamientos del robot utilizado. Seleccionaremos el fichero ***Rug_Warrior_Pro.icd***



6) Aparecerá una nueva ventana de diálogo donde se nos pide confirmar la ubicación correcta del conmutador en la posición Download. Confirmaremos pulsando el botón ***Ok***, comienza de este modo la descarga (*el LED amarillo parpadea*). En primer lugar descarga los 256 bytes iniciales que espera el 68HC11 y comienza su ejecución. El código almacenado en estos 256 bytes es una cargador que se encarga de transferir el S.O. a la memoria RAM externa (32 KB), por lo que seguirá transfiriendo datos por el puerto serie.

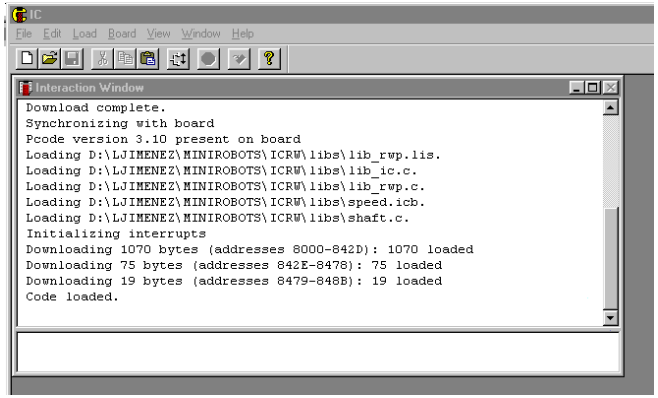


7) Si todo es correcto deberá salir el siguiente mensaje "***Dowload Complete: please switch up to the Run position before clicking Continue***". Si hubiera habido un error repetiríamos de nuevo los mismos pasos. (En ese caso debemos resetar previamente la tarjeta mediante el pequeño botón en la esquina inferior izquierda)



8) Haciendo caso del mensaje colocaremos el conmutador en la posición superior (***Run***). En la pantalla LCD debe aparecer un mensaje de bienvenida y un pequeño corazón parpadeante en la esquina inferior derecha. Este corazón nos indica el estado del sistema operativo. Si no parpadea es que se ha *colgado*.

   Pulsamos Aceptar y comienza a cargar las librerías específicas del robot tal como aparecen en la imagen siguiente.



En este instante ya está completamente arrancado el programa IC tanto en el PC como en el Robot. Aparece una ventana con dos partes:

- La parte superior nos muestra los mensajes asociados a las operaciones que realizamos (ventana informativa)
- La ventana inferior nos permite teclear comandos de control del robot.

Podemos probar en este momento el interprete de comandos:
Tecleamos : `printf("Hola")`

En la pantalla LCD debe aparecer el mensaje indicado. En la ventana de mensajes del entorno aparecerá el proceso de descarga del código y el valor devuelto por la función:

```
IC> printf("Hola")
Downloading 17 bytes (addresses C200-C210): 17 loaded
<int> 0
```

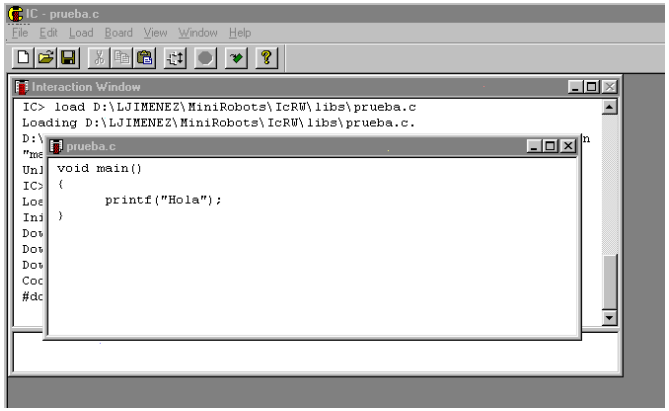
3.2 El primer programa en IC

Vamos ha realizar nuestro primer programa. Será muy sencillo y únicamente visualizará una cadena de texto en pantalla. El objetivo es familiarizarlos con el editor y la descarga de programas.

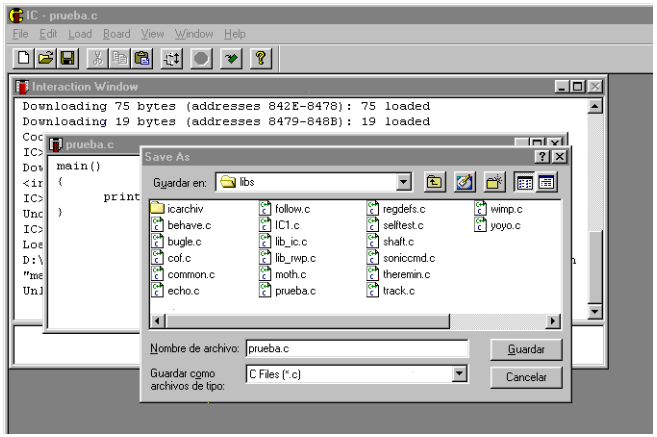
Nota: se supone que se conoce la programación en C

Como en todo programa en C el código se ubica en funciones. Entre ellas existirá una que se denomina ***main()*** y que es la que se ejecuta cada vez que se enciende o se reinicializa la tarjeta controladora (la función no devuelve ningún parámetro *void*, ni se le puede pasar ninguno)

Seleccionaremos la opción del menú ***File/New***: Nos aparecerá una ventana con un editor de texto sencillo. Escribiremos el siguiente código



Seleccionaremos la opción ***File/Save*** para almacenar el código C en un fichero denominado prueba.c. Lo ubicaremos dentro del subdirectorio *c:\ic\libs*.



Seleccionaremos la Opción del menú *Load/Download Window*(descarga el programa de la ventana actual. Igualmente se podría usar la opción *Load/Download File* seleccionado el fichero que hemos almacenado.

En la ventana de mensajes nos indica si la compilación fue correcta y el estado de la descarga en la tarjeta. Si existiera algún error de sintaxis nos lo indicaría. Debiendo corregirlo y proceder a descargarlo de nuevo.

Si la descarga tiene éxito bastará que pulsemos el botón de ***reset*** para que se ejecute el programa.

3.3 Utilizando los sensores y actuadores

Ya conocemos como editar y descargar un programa. Vamos ha realizar ahora un programa sencillo que utilice los sensores y actuadores del robot.

Realizaremos un programa que lea continuamente la señal del micrófono. Cuando ésta señal supere un cierto umbral (golpe) activará los motores para que avance el robot. Un nuevo golpe parará los motores.

Para ello debemos usar funciones que nos permitan leer la señal de altavoz y suministrar corriente a los motores (*ver el manual de IC para los detalles de cada función*).

La primera de ellas es la función ***analog(int canal)*** que nos permite leer el valor muestreado por uno de los 8 canales analógicos disponibles (rango 0-255). En nuestro caso el micrófono está conectado al canal **2**. Como la señal leída fluctúa según la frecuencia del

sonido tomaremos la diferencia absoluta respecto al valor medio (128), este valor es comparado con un umbral.

Para activar los motores disponemos, entre otras, de la función ***drive(int t, int r)***. El primer parámetro especifica la velocidad traslacional y el segundo la velocidad rotacional. Según el valor de los parámetros se determina la energía suministrada a cada motor (modulación en ancho de pulso). A continuación se muestra el código que teclearemos y cargaremos en el robot. Probablemente tengamos que ajustar manualmente el valor del umbral de sonido.

```
/* Control del robot mediante sonido */

int MICRO       = 2;      /* Canal del micrófono*/
int umbral_sonido= 40;          /* nivel de sonido reconocido */


void main()
{
  int nivel= 0;           /* nivel leido del micrófono */
  int motores =0;         /* indica el esto activado o desactivado de los
motores */

  printf("Sonic Commander");

   /* ejecuta de forma continua */
  while (1)
  {

      nivel = abs(analog(MICRO) - 128);/* diferencia respecto al val. medio */
      if ( nivel > umbral_sonido)
      {
         if (motores == 0)
         {
             motores=1;   /* motores activados */
             drive(100,0);/* velocidad traslación 100: velocidad giro: 0 */
         }
         else
         {
             motores=0;   /* motores desactivados */
             drive(0,0);  /* velocidad traslación 100: velocidad giro: 0 */
         }
      }
      msleep(100L);/* espera 100 ms*/

  }
}

int abs(int arg)
{  if (arg < 0)
      return -1 * arg;
    else
      return arg;
}
```

Nota: si tenemos cargado el programa previo en el Robot nos dará un error al realizar la descarga ya que la función main() estaría repetida. Debemos borrar el programa previo con el comando ***unload prueba.c*** (aquí debéis poner el nombre del fichero a borrar), que ejecutaremos en la ventana inferior.

3.4 Programación Multitarea

Veamos a continuación las posibilidades que tiene IC par ejecutar varias tareas de forma concurrente. Para ello tomaremos el código anterior y lo incorporaremos como una tarea. Adicionalmente incorporaremos una nueva tarea que leerá el estado de los **bumpers** (tres interruptores de contacto que rodean al robot) y lo mostrará en pantalla.

La función para crear un proceso es ***start_process(function-call(...),[TICKS],[STACK-SIZE])***. El concepto de proceso IC es equivalente al de thread ya que todos comparten un espacio de datos común. El primer parámetro es obligatorio e indica la función de inicio del proceso. Adicionalmente se puede especificar el tiempo de ejecución y el tamaño del stack (si no se especifican se asignan unos valores por defecto)

```
/* Control del robot mediante sonido - Multitarea */
int MICRO       = 2;      /* Canal del micrófono*/
int umbral_sonido= 40;          /* nivel de sonido reconocido */

void main()
{
      printf("Sonic Commander\n");
      start_process(sonido());
      start_process(bumpers());
}

void sonido()
{
  int nivel= 0;     /* nivel leido del micrófono */
  int motores =0;  /* indica el esto activado o desactivado de los motores */

  while (1)
  {
      nivel = abs(analog(MICRO) - 128);/* diferencia respecto al val. medio */
      if ( nivel > umbral_sonido)
      {
         if (motores == 0)
         {  motores=1;   /* motores activados */
            drive(100,0);/* velocidad traslación 100: velocidad giro: 0 */
         }
         else
         {  motores=0;   /* motores desactivados */
            drive(0,0);  /* velocidad traslación 100: velocidad giro: 0 */
         }
      }
      defer();     /* libera el uso de CPU */
  }
}

void bumpers()
{
      int bump_stat;

      while (1) {
            bump_stat=bumper();
            printf("Bumper: %d\n", bump_stat);
            sleep(1.0);
      }
}
/*Incluir sólamente si no se carga comun.c
 int abs(int arg)
{  if (arg < 0)
      return -1 * arg;
    else
      return arg;
}*/
```

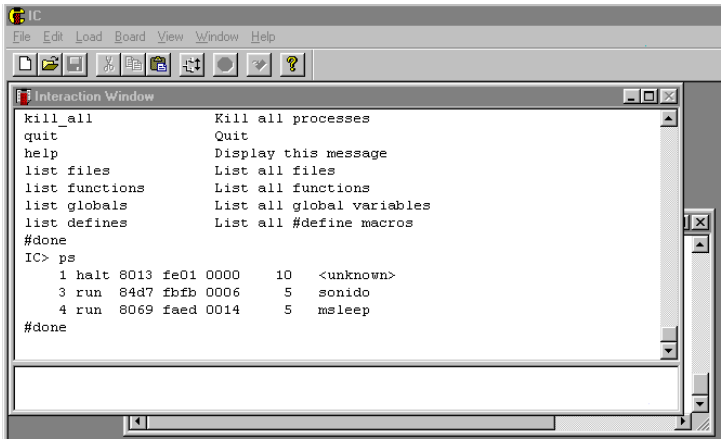

El planificador es de tipo ***cuantum*** fijo, por lo que los procesos se van alternando en el uso de la CPU, siendo interrumpidos cuando expira su cuantum.

Nota: la función ***defer()*** libera el uso de la CPU. De esta forma nos aseguramos de que la próxima vez que le toque al proceso el uso de la CPU estará al principio del bucle *while()*, garantizando que no se interrumpa a la mitad. La función ***sleep(1.0)*** establece un retardo de 1 segundo en la lectura de los bumpers.

3.5 Utilizando el Depurador

El entorno integrado dispone de un depurador que nos permita conocer el valor de la variables globales en tiempo de ejecución, basta con teclear el nombre de la variable.

Asimismo podemos conocer los procesos actualmente en ejecución con el comando ***ps***, así como todas las funciones cargadas (***list functions***)



3.6 Demostración de las posibilidades del RugWarrior.

Para finalizar esta primera práctica, y a modo de introducción de lo que conseguiremos al acabar este curso, vamos a cargar un conjunto de programas de demostración disponibles en el entorno IC.

Descargaremos las demos mediante el fichero ***behave.lis***. Este fichero se puede encontrar en el subdirectorio **/libs** (en algunos equipos puede encontrase en el subdirectorio **/libs/demo**).

Los ejemplos se van ejecutando de forma secuencial al ir pulsando el botón de reset, podremos observar diferentes comportamientos en función de los sensores disponibles. No te preocupes por ahora de comprender el código de estos programas, poco a poco avanzaremos en la programación de la ***Inteligencia Artificial*** del robot, por ahora simplemente disfrútalos.